Cleared for open publication by SDIO 11 June 87
EZa

APPROVED FOR
PUBLIC
RELEASE

Research for Adaptive, Distributed Network Management System

NRL Contract No. N00014-86-C-2056

20040817 053

HARRIS SIMULATOR DESIGN
DESCRIPTION

CDRL A003

U1199

ACN-005-87
cy1

HARRIS

HARRIS CORPORATION GOVERNMENT SPECIAL PROGRAMS OPERATION
P.O. BOX 95000, MELBOURNE, FLORIDA 32902, (305) 729-7401

U 01199

Harris Simulator Design Description

for

Adaptive Distributed Network Management System

Prepared for:

Naval Research Laboratory
Contract No. N00014-86-C-2056
CDRL A003

Submitted By:

Harris Government Special Programs Operation
P.O. Box 95000
Melbourne, FL 32902

5 September 1986

The Contractor, Harris Corporation, Government Systems Sector, hereby certifies that, to the best of its knowledge and belief, the technical data delivered herewith under Contract No. N00014-86-C-2056 is complete, accurate, and complies with all requirements of the contract.

T.E. Kryst

T.E. Kryst, Engineer
Special Programs Operation

[illegible]

P.J. Knoke, Program Manager
Special Programs Operation

[illegible]

C.L. Mohre, Director Engineering
Special Programs Operation

HARRIS SIMULATOR DESIGN DESCRIPTION

TABLE OF CONTENTS

HARRIS SIMULATOR DESIGN DESCRIPTION

TABLE OF CONTENTS (Continued)

HARRIS SIMULATOR DESIGN DESCRIPTION

TABLE OF ILLUSTRATIONS

1. INTRODUCTION

1.1 Identification and Purpose

This design document satisfies Contract Data Requirements List #A003 of contract No. N00014-86-C-2056, Adaptive Distributed Network Management System (ADNMS), Naval Research Laboratory (NRL).

The document describes the Harris Simulator used to support the development and test of a first generation network management algorithm for a typical SDI communications network. The communications network is superimposed upon an SDI "architecture" (configuration of battle management, sensor, and weapons platforms) which was provided as GFI. The SDI communications network is assumed to be a packet switching network.

1.2 Overview

The Harris simulator may be regarded as an algorithm testbed. The primary requirement driving the design of this testbed is the need to provide a reasonably realistic SDI-oriented environment while avoiding excessive detail. To minimize simulator development cost and schedule requirements, existing software modules have been used wherever appropriate.

Section 2 provides an initial set of simulator requirements. These requirements are still in the process of evolution and refinement. The Concept of Operation (Figure 2.2) specifies major system functions and illustrates the flow of control in a simulation experiment. The system functional decomposition shown in Figure 2.2 supports the design goal of significant software module reuse.

Section 3, Top Level Design, describes how the functions identified in Section 2 have been mapped into specific software components. Where existing software is incorporated or modified for use within the system, that software is described briefly. More details can be obtained from the references in Section 5. Two Top Level Computer Software Components (TLCSC's) are discussed in this section, namely the Communications Configuration Generator and the Message Script Generator.

Section 4 is dedicated to the description of a third TLCSC, called the Comm Network Component Simulator. This simulator provides a networking environment for communications Algorithms Under Test (ALGUT's). This simulator uses the ISO/OSI layered network design concept. Under this concept, networking functions are defined and allocated to one of seven OSI layers. Interfaces between layers are well defined. The use of this design concept facilitates the merging of the Harris Simulator implementation with existing NRL testbed code.

Acronymns used in this report are each defined at the point of first use. Table 1.2 groups all acronymn definitions in one place as a convenience to the reader, and also defines some other technical terms which might be ambiguous.

Table 1.2 Terms and Abbreviations

ADNMS	- Adaptive, Distributed Network Management System
ALGUT	- Algorithm Under Test
Analyst	- Person who sets up and runs simulation experiment
AR	- Adaptive Routing
ARM	- Adaptive Routing Manager
ARQ	- Automatic Repeat Request
BM	- Battle Management
BM/C^3	- Battle Management/Command, Control and Communications
Bus	- That portion of an ICBM which contains the RVs and decoys
C^3	- Command, Control and Communications
CSCI	- Computer Software Configuration Item, the top most level design module
EMP	- Electro-Magnetic Pulse
ESD	- Electronic Systems Division, U.S. Air Force
GFI	- Government Furnished Information
ICBM	- InterContinental Ballistic Missile
ISO	- International Standards Organization
KEW	- Kinetic Energy Weapon
KEWSTAR	- An existing threat simulation on which portions of the design are based
Layer 1	- An existing set of NRL SIMULA code used to simulate a generalized network system
LSITES	- Launch Sites
MOP	- Measure of Performance
NRL	- Naval Research Laboratory
OSI	- Open Systems Interconnect
Phase 1	- The planning phase of a Simulation experiment
Phase 2	- The execution phase of a Simulation experiment
Phase 3	- The data analysis and evaluation phase of a Simulation experiment
RC	- Resource Configuration (synonymous with Link Assignment)
RCM	- Resource Configuration Manager
RV	- Reentry Vehicle (warhead)
SDI	- Strategic Defense Initiative
SPSS	- Statistical Package for Social Scientists
TLCSC	- Top Level Computer Software Component
TSITES	- Target Sites

2. REQUIREMENTS

2.1 General Requirements

1. All simulation software must be compatible with and implemented on the Sun workstation.

2. All data used for simulator development and experimentation must be derived from unclassified sources.

3. The Simulator used to represent a communication network environment must be implemented in SIMULA to assure compatibility with existing NRL software. Also, contexts and interfaces must be compatible with the existing NRL Layer 1 simulation.

2.2 Functional Requirements

The Harris simulator consists of a set of five loosely coupled functions. The five functions fall into the three categories of Simulation Experiment Planning, Simulation Experiment Execution, and Simulation Output Data Analysis. Figure 2.2 shows the functions in each category with arrows indicating system functional data flow.

Each of the five functions is described in the following subsections. The descriptions include statements of function, purpose, input data, and output data.

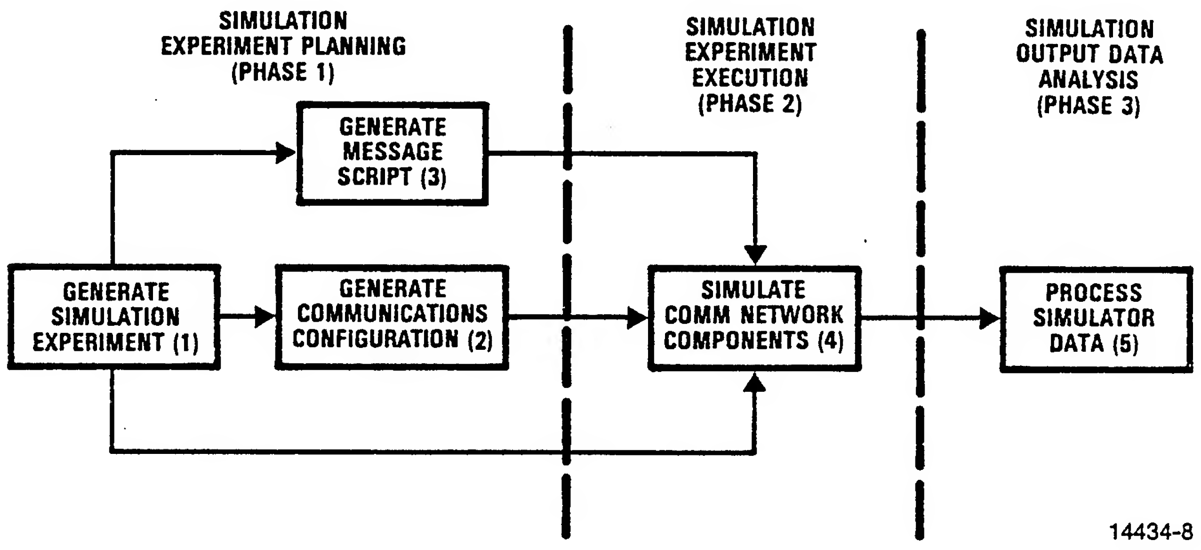

SIMULATION EXPERIMENT PLANNING (PHASE 1)
SIMULATION EXPERIMENT EXECUTION (PHASE 2)
SIMULATION OUTPUT DATA ANALYSIS (PHASE 3)
GENERATE MESSAGE SCRIPT (3)
GENERATE SIMULATION EXPERIMENT (1)
GENERATE COMMUNICATIONS CONFIGURATION (2)
SIMULATE COMM NETWORK COMPONENTS (4)
PROCESS SIMULATOR DATA (5)
14434-8

Figure 2.2 - Harris Simulator Concept of Operations

2.2.1 Generate Simulation Experiment

Purpose: Provide the means to manipulate the files containing the data required to define and control a simulation experiment.

Input Data: A set of prewritten or standard files provides the bulk of the input data. Some data is provided by interactive user input to tailor an experiment. There are several types of input data to the Simulator for experiment definition. These types include:

- Raw script and configuration data as described in Sections 2.2.2 and 2.2.3.
- Other model parameters needed to describe submodels, including link, node and message flow descriptors.
- Experiment parameters that describe a series of model executions.
- Other controls in the form of software switch settings that dictate the statistics to be collected and/or the debugging values to be output.

Output Data: A file for the particular experiment about to be run.

2.2.2 Generate Communications Configuration

Purpose: Provide a timeline of predictable link, node, and inter-nodal distance changes, due to dynamic orbital relationships. Also, determine link and node losses and degradations due to threats to the SDI shield (e.g., jamming and EMP). As an input to the simulation, the communications configuration determines network connectivity changes versus time.

Input Data: Data describing the communication network physical architecture, and the shield threats to be modeled. The data includes:

- Node orbit descriptions
- Relative positions of the communications network nodes
- Initial positions of communication network nodes and the earth
- ASAT and EMP-induced node outages
- Link outages due to jamming, scintillation, etc.

Output Data: Conveys the time of each event, the event name, and a list of event descriptor data. Events include position changes and link and node destruction/degradation

It is assumed that nodes do not operate at partial capacity; i.e., they are assumed to be either completely destroyed or completely operational. In the case of a memory wipe due to a nearby nuclear blast (EMP), the node outage may be only temporary.

2.2.3 Generate Message Script

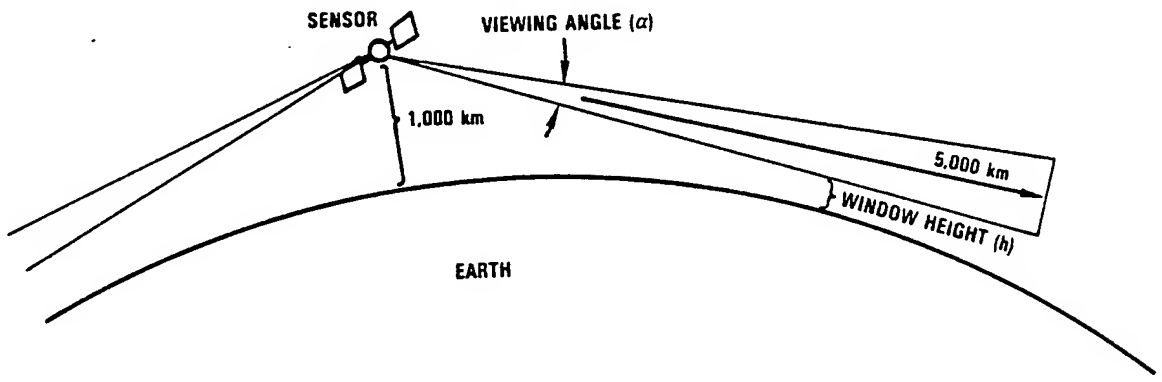
Purpose: Construct a timeline or "script" of sensor observations. The script should be traceable to the selected ballistic missile threat model.

Input Data: Data on sensor characteristics and ICBM threat characteristics. Sensor parameters are depicted in Figure 2.2.3.

Aside from orbital parameters, the sensor submodel must have parameters for the window height, viewing angle and range as illustrated in the figure. That capability allows for incorporation into the simulation of existing and projected technology. At present, window height must be 100-200 km because of dispersion effects of the atmosphere. The viewing range is assumed to be 5000 km.

The ICBM threat must be simulated in sufficient detail to ensure reasonably accurate sensor sighting reports. A spike attack is assumed to be most stressing for communications purposes. The missile launch geographical distribution and distribution of destination sites must be representative samples of known and expected sites, in order to model realistic missile clouds.

Output Data: Indicates the active sensors and the number of threat objects viewed by each active sensor for selected time periods.



14395-3

Figure 2.2.3 - Midcourse Sensor Submodel

2.2.4 Simulate Comm Network Components

Purpose: Simulate the communications network components at levels of detail appropriate for an accurate study of adaptive routing and resource configuration algorithms within selected SDI operational scenarios (including threat scenarios).

Input Data:
a. Script generator data (sensor messages)
b. Communications network configuration data
c. Other input parameters that may be needed to describe the communications network (e.g. link bandwidth, node data storage and processing capabilities, etc.)

Description of Components: Algorithm simulation requires that components of the SDI communications network be modeled to at least the level of detail needed to support the network management algorithm study. These anticipated components are nodes, links, messages, decision functions, and node recovery functions. These five components are discussed below.

1. Nodes: Each node must communicate with adjoining nodes at various layers of the ISO/OSI model. The degree of detail or fidelity required for communications at the seven layers is estimated below.

Layer #	Layer Name	Degree of Detail
1	Physical	Medium-High
2	Data Link	Low-Medium
3	Network	High
4	Transport	High
5-7	Session, Presentation & Application	Low

Layers 3 and 4, the network and transport layers, require much detail because they contain most of the network management functions of interest. Layer 1, the physical layer, is also important because it includes functions which handle the changing topologies due to orbital dynamics, and functions which handle antenna pointing. Layer 2, the data link, is specified only to the degree necessary for a detailed routing algorithm implementation. Layers 5 and 6 are unimportant for purposes of this research. Layer 7 functions are important only with respect to the quantity and patterns of messages generated within nodes.

Three types of nodes are assumed for the SDI Communications network, namely sensor, weapon, and battle management nodes. The main internodal communications traffic flow for the simulator is:

- Sensors - Accept the sighting reports put out by the message script generator, and send corresponding sensor messages to battle manager (BM) nodes
- Weapons - Accept firing orders from the BM's via weapons allocation messages.
- Battle Managers - Process sensor messages and send associated firing orders to the weapons via weapons allocation messages.

2. Links: Characterize the channel between communicating nodes. Links may be created or destroyed temporarily or permanently. They always have propagation delay. Link types to be modeled include point-to-point, broadcast and limited broadcast. Point-to-point links may be multiplexed. Each node is assumed to have 3-5 physical links, with any combination of link types possible for a particular experiment.

3. Messages: Represent all possible data that is transmitted between nodes. At least the following SDI Communication message attributes must be explicitly modeled:

Identifier	Priority
Time originated	Quality of service required
Time sent on this link	Source
Deadline for delivering msg.	Destination
Message type	Message length
Message subtype	Number of hops up to this point
	Max permissable number of hops

4. Decision functions: Decision functions are application layer functions representing battle management or routing actions that cause delays, react to message receipt, send messages, and gather information. Requirements on decision functions are TBD.

5. Node Recovery: EMP effects upon the nodes are assumed to be temporary. It is further assumed that an internal node function exists that periodically saves internal databases on a nonvolatile storage medium. In the case of an EMP event, all node data is lost including buffers, except for information previously saved on nonvolatile storage.

Output Data: Consists of raw communications network performance data, attributable to the ALGUT. This data may be broken down in various ways, e.g. by messages and types, links and priorities, etc. The measurements required are described in greater detail in Section 2.2.5 (Process Simulator Output Data).

2.2.5 Process Simulator Output Data

Purpose: Provide summarized simulation output in the form of reports, tabulations, and graphs. Conduct statistical tests using the summarized simulation output to evaluate the performance of each ALGUT.

Input Data: Measures of performance (MOPs) that apply to the simulation experiment. The MOPs are used to compare alternative algorithms implementing the same function. MOPs must be provided as needed for an experiment. The MOPs to be provided include but are not limited to:

- o Average and variance of message delivery delay
- o Probabilities of message delivery, and of message delivery before a deadline
- o Communication link utilization per link and aggregate
- o Communications overhead required for algorithm execution
- o Algorithm time and storage requirements
- o Number of reasonable disjoint routes available between source and destination
- o Reconfiguration update transient time

Output Data: Statistical charts, graphs and tables as needed to analyze system performance.

2.3 Interface Requirements

2.3.1 Interface Identification: Seven interfaces between major system functions have been identified. Figure 2.3 is a functional flow diagram from Figure 2.2 in which the network management algorithms to be tested (ALGUT's) are also shown. Major interfaces are named and illustrated as dashed boxes.

Table 2.3 summarizes information passed across each interface. Each interface is identified by number and name, followed by a brief description of the data passed.

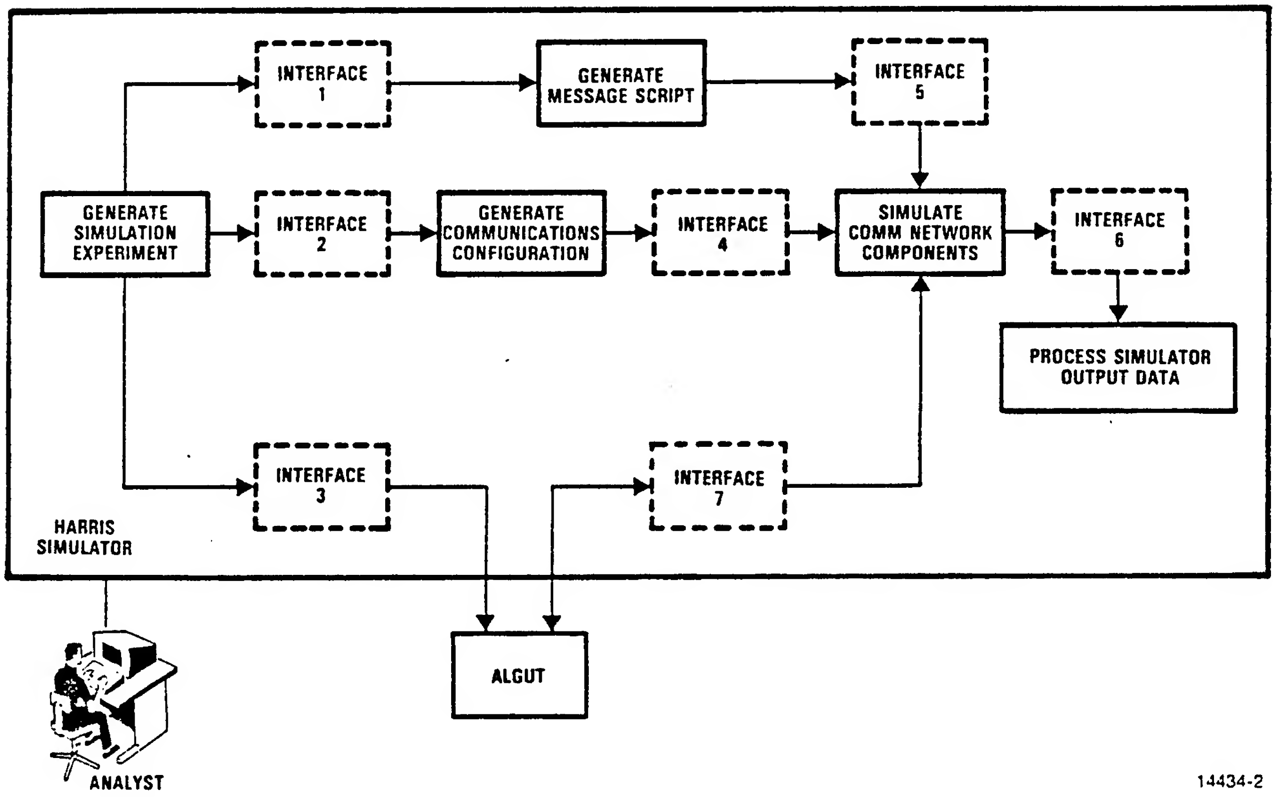


Figure 2.3 - Interface Block Diagram

I/F #	I/F Name	Information Passed Forward	Information Returned
1	GSE/GMS	Description of Modeled Threat Orbital Configuration Sensor Parameters	None
2	GSE/GCC	Orbital Configuration Start Parameters	None
3	GSE/ALGUT	ALGUT specific parameters	None
4	GMS/SCC	Time and number of threat sightings for each sensor	None
5	GCC/SCC	Time of update Updated position coordinates for each satellite Node and link status updates	None
6	SCC/PSD	Performance data	None
7	ALGUT/SCC	Internal node status Network status updates	Antenna control instructions Routing table updates Status requests

Table 2.3 - Interface Summary Table

2.3.2 Interface Discussion: Since none of the five major testbed components pictured in Figure 2.3 must run concurrently, there are few interface requirements. There are some general guidelines, however:

- o All information transfers should take place within the same SUN workstation where possible, to insure compatibility with NRL and streamline the analyst tasking.
- o Ease of use is assumed to be the guiding factor in interface construction. However, in the interests of speed, an initial simpler version may relax this guideline.
- o Standards are to be devised and used uniformly for information transfer.

The ALGUT, which is not technically a testbed function, executes concurrently and synchronously with the component simulator. Interface 7 provides all communications capability needed by the ALGUT, and is the most critical. Because of this, the following additional requirements are appropriate to interface 7. It must:

- o Assume a "black box" set of functions that is entirely separate and distinct from testbed code.
- o Provide for the following two major network management functions:
  - Adaptive Routing, which determines which link and node sequence to use for forwarding messages in a dynamic environment.
  - Resource Configuration*, which determines how to enable and disable links for efficient operation in a dynamic environment.
- o Be generalized so as to accommodate all foreseen information flow for the above two functions.
- o Use existing interface definitions of the present NRL code, expanding upon them where necessary.

* Resource Configuration is now called Link Assignment

3. TOP LEVEL DESIGN

This section provides a top level view of the major Harris simulator components defined in the requirements section. A description of the Computer Software Configuration Item's (CSCI's) is given, along with important lower level relationships with TLCSC's and other modules. Section 4 is devoted to a detailed description of the Communications Network Component Simulator.

3.1 Allocation of Functions to CSCI's

The Concept of Operations (Figure 2.2) is the basis for the definition and development of the CSCI's. Table 3.1 lists the functions in the concept and the corresponding CSCI's in the design which accomplish those functions. The five functions of the Concept are mapped into three CSCI's and two other tasks. The other tasks, Generate Simulation Experiment and Process Simulator Output Data functions, are not fully integrated into this initial simulator design. The Process Simulator Output Data function utilizes an off-the-shelf statistical package called SPSS for evaluating experiment results.

Three CSCI's are used to perform the remaining functions. The Script Generator produces a timeline of sensor sighting observations. The Configuration Generator provides information on spatial relationships between the nodes, and also information on node and link outages. The Component Simulator provides a plug-in environment for the ALGUT.

The following subsections provide a high-level description of the CSCI components and their intercommunications.

CONCEPTUAL FUNCTION	CORRESPONDING CSCI		REMARKS
	CSCI #	CSCI NAME	
Generate Simulation Experiment	-	---	Manual task in the initial version of the Harris Simulator
Generate Message Script	1	Script Generator	Generates numbers of sighted targets per sensor vs. time
Generate Comm Configuration	2	Configuration Generator	Generates relative positions of participating nodes as well as node and link outages
Simulate Comm Network Components	3	Component Simulator	Provides an environment for investigating ALGUT performance
Process Simulator Output Data	-	---	Semi-manual task in the initial Simulator. Utilizes existing statistical tools (SPSS)

Table 3.1 - Operational Function to CSCI Mappings

3.2 CSCI Architecture

The Configuration Generator and Script Generator are both contained within a single modified program. That program operates independently of the Component Simulator, communicating with it only via file transfers. Overviews of the Generators are given in the following paragraphs. The Component Simulator, the key CSCI, is described in Section 4. Figure 3.2 summarizes the CSCI architecture.

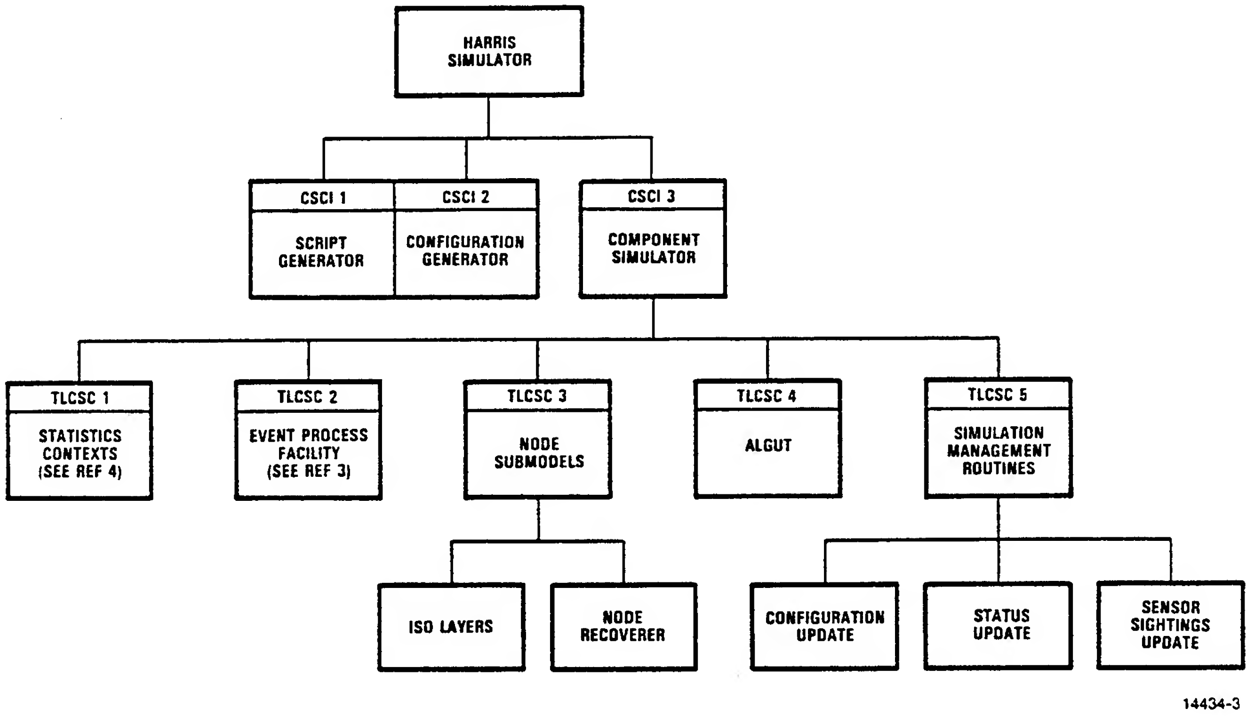


Figure 3.2 - CSCI Architecture

The Component Simulator consists of five elements, namely TLCSC's 1 to 5. TLCSC 1 (Statistics Contexts) and TLCSC 2 (Event Process Facility) consist of generalized facilities that will be used for statistics collection and process scheduling, respectively. TLCSC 3 (Node Submodel) is compatible with existing NRL code, most of which is described in Ref. 2. TLCSC 4 (ALGUT) is included for convenience, but it is not technically a part of the Harris Simulator. TLCSC 5 (Simulation Management Routines) includes routines of a general purpose that are used to drive the simulation and to complete the interface with CSCI's 1 and 2.

3.3 Functional Control and Data Flow

Figure 3.3 illustrates operational control and data flow between CSCI's. Two files, the Component Parameters file and the Status file, are tailored for the experiment manually by the analyst (i.e., the person who sets up and runs the Simulation experiment). The site files (Launch Sites and Target Sites, which control coordinates needed to describe the ballistic missile threat model), are considered to be fixed for the purposes of this study.*

The Script Generator and the Configuration Generator require orbital and experiment parameters which are set up by the analyst interactively. After execution of the Script and the Configuration Generators, all the files needed for the simulation experiment are ready.

The simulation experiment proceeds with a set of simulation executions which use the four files just discussed. It is expected that many executions will be made of the simulator for a single set of script and configuration files. Output data is gathered as needed. Finally the simulation experiment proceeds to the final phase, analysis and evaluation of the output data.

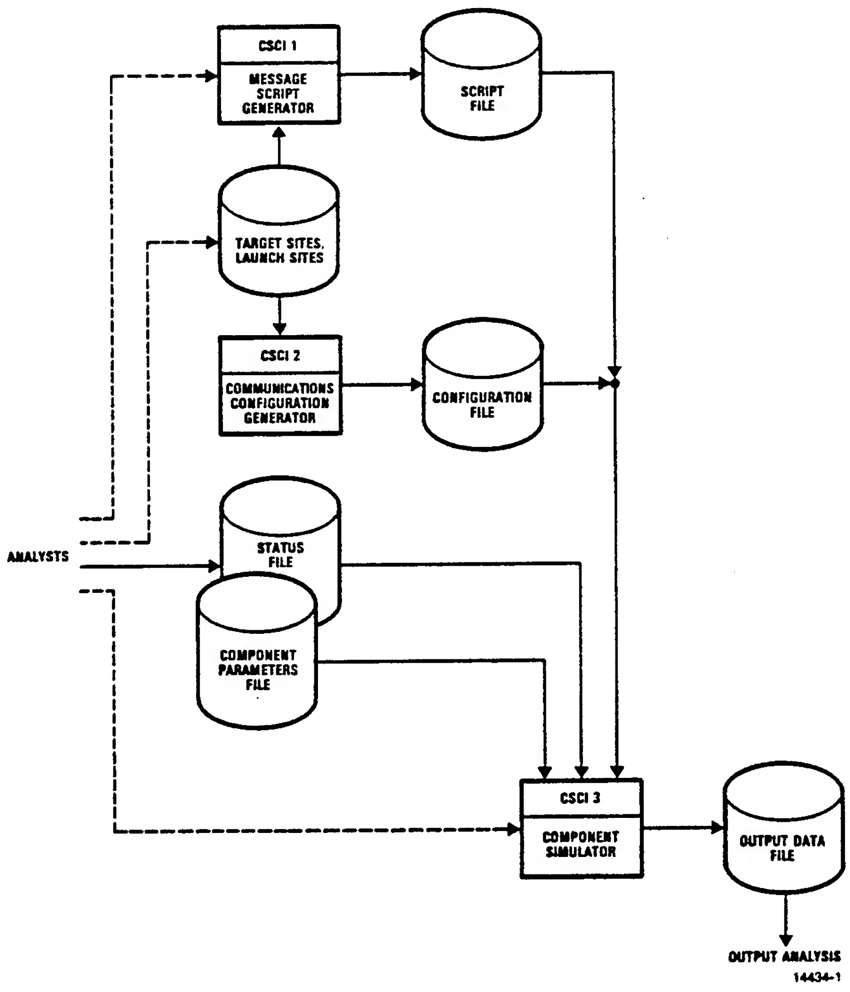


Figure 3.3 - CSCI Functional Control and Data Flow

* However, it is not difficult for a person very familiar with Harris Simulator code to change the site files, and therefore the nature of the ballistic missile attack.

3.4 Communication Configuration Generator

3.4.1 Introduction

An existing threat model (KEWSTAR, Ref. 5) is used to satisfy the bulk of configuration generator requirements. This threat model was originally developed to study KEW effectiveness, but it contains many modules useful to the testbed. Satellites are positioned in space and their coordinates are updated as time progresses. Equations for orbital updates have been implemented using Bate's derivations (Ref. 7). A constellation of similar orbits is described by the user via parameters.

3.4.2 Ballistic Missile Threat

In a similar manner, the ICBM threat is also described with parameters denoting missile launch duration and number of missiles. The launch distribution and parameters for the missile trajectories have been approximated from unclassified sources, sometimes by working backwards from known data. Five types of ICBM have been included in the data, and provision has been made to add other types when data is available. Table 3.4.2 shows the five ICBM types as well as the proportionate number of each type assumed to be included in an actual attack. A variable total number of attacking ICBMs can be handled by the simulator, but the percent mix shown is maintained constant.

Launch and destination ICBM sites have been derived from unclassified sources. A simplification narrows the number of sites to just 19 known launch sites distributed throughout the Soviet Union, and 20 airbases and cities in the United States.

ICBM Type	Number of ICBMs	%
SS-18	308	22
SS-19	330	24
SS-17	150	11
SS-13	60	4
SS-11	550	39
TOTAL	1398	100

Table 3.4.2 - Assumed Soviet ICBM threat distribution.

3.4.3 Threats to the SDI Shield

The above ballistic missile threat model is convenient for satisfying most Configuration Generator requirements. An additional function is needed, however, to model the threat to the deployed communications network nodes and links (i.e. the threat to the SDI "shield" itself).

The additional requirement is satisfied with the addition of a node/link status file. By this means, the effects of EMP and of node and link destruction due to nuclear bursts as well as other ASAT and jamming effects can be included in the simulation. The user can formulate any random or nonrandom set of outages desired. The communications configuration status file organization is depicted in Table 3.4.3. Each time an event causing link or node outages occurs, a list is made of nodes and links affected, with each identifier on a separate line. The node or link is specified as enabled or disabled at that time, and a note is made regarding whether or not the outage is temporary. If it is temporary, the simulator will automatically recover the node according to the EMP recovery time parameter.

DATA ITEM	DENOTES
Time	When event occurs with respect to start of simulation
Node/Link ID	Unique node/link name
Type Event	Outage or return to service
Type of Outage	Temporary (EMP-caused) or Permanent

Table 3.4.3 - Status File Data Description

3.4.4 General KEWSTAR* Changes Made

KEWSTAR needed several minor modifications to satisfy ADNMS requirements. Three changes that were made are listed below with rationale.

o Constellation Enhancement - In KEWSTAR, a single constellation of platforms is assumed, all with the same altitude, shape and inclination. The ADNMS problem, however, requires at least three different constellations as shown in Figure 3.4.4. Code existed within KEWSTAR to compute the needed constellations, but it had to be expanded to allow for the simultaneous use of the three assumed sets of satellite orbits.

o Configuration data - Constellation configuration data needed by ADNMS was produced and used internally by KEWSTAR, but it was not explicitly output by the program. Appropriate statements were added to KEWSTAR to communicate satellite position data into an intermediate file to be used by the Harris Simulator.

o Other modifications - KEWSTAR contained code that was not needed for the Harris Simulator. To improve efficiency, some of that code was removed.

3.4.5 Inputs

Parameters required by the Configuration Generator are obtained from three sources: the TSITES file, the LSITES file, and the interactive input. TSITES contains coordinates and identifiers for ICBM targets. LSITES contains missile data and launch site coordinates. The user may specify threat, orbital and other parameters interactively.

* KEWSTAR is the basis for the Script Generator and the Configuration Generator. See Reference 5.

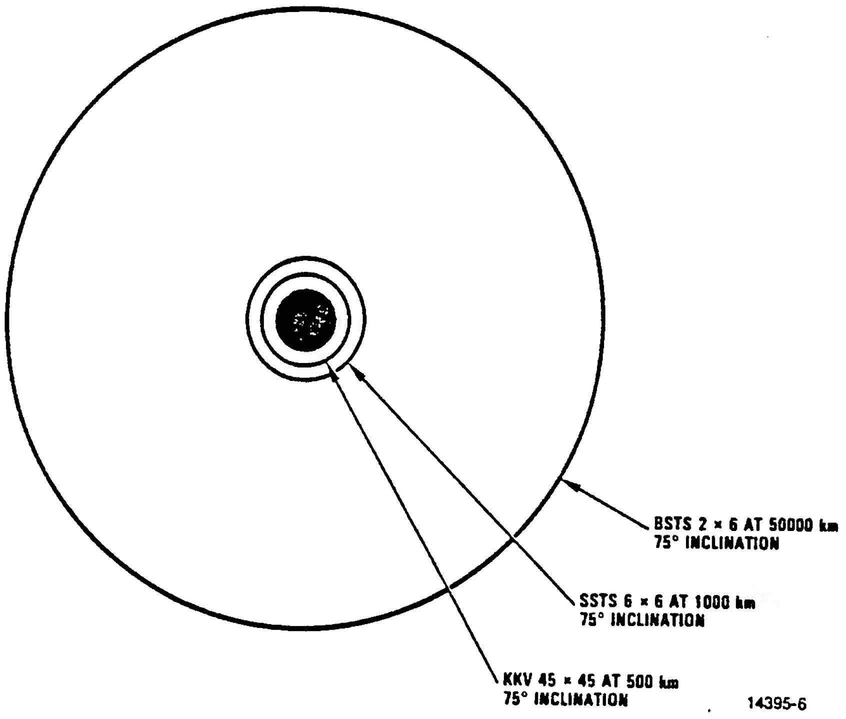


Figure 3.4.4 - GFI SDI Architecture

3.4.6 Outputs

The Configuration Generator produces two files: the Configuration File and the Status File (discussed previously). The Configuration File consists of position coordinates for each platform in a three dimensional cartesian coordinate system, at time-stepped intervals. Table 3.4.6 shows the file organization. The file is used by the simulator to update the relative positions of satellites and to track spatial relationships.

DATA ITEM	DENOTES
Time	Reference, from start of simulation, when position occurs
Node ID	Unique node name
X	First cartesian position coordinate
Y	Second cartesian position coordinate
Z	Third cartesian position coordinate

Table 3.4.6 - Configuration File Organization

3.5 Message Script Generator

3.5.1 Introduction

The Message Script Generator is designed to be a stand alone driver for the Simulator. Since a major stress on the SDI communications network is the massive volume of sensor messages that must be sent during an attack, the number and pattern of the sensor sightings which generate those messages is critical. Because the sensor sightings have a significant effect on the communication network message patterns and on resulting performance, it is desirable that realistic sighting patterns be generated. Modified KEWSTAR (KEWSTAR/M) satisfies the requirements for realism since it has a capability to generate reasonable approximations to missile trajectories and relative sensor satellite positions.

3.5.2 KEWSTAR/M

Five additional minor adjustments were made to KEWSTAR to satisfy ADNMS requirements. Each needed change is discussed below with rationale.

o Missile trajectories - KEWSTAR was originally designed primarily as a boost phase model. While equations are included for the midcourse phase of flight, that phase is deemphasized. Examination has shown that some adjustment of the KEWSTAR trajectories was needed, since boost phase exit velocities caused the warheads to be somewhat higher and travel farther than is realistic.

o Missile launch distribution - No modification was made to the existing launch distribution implementation, because a spike type of attack is assumed to be most stressing for communications. However, one or more additional attack scenarios may eventually have to be incorporated into KEWSTAR/M to verify that ADNMS algorithms operate well under various ballistic missile attack scenarios.*

o Sensor submodel - KEWSTAR assumes a perfect sensing capability. This is unsatisfactory for ADNMS purposes. A submodel representing the mechanics shown in Figure 2.2.3 was incorporated into the model for computing sensor-missile sighting patterns.

o Threat assumptions - KEWSTAR distributes the five modeled types of ICBMs among existing bases according to known basing information. That level of detail is excessive for ADNMS purposes. One simplifying assumption made for ADNMS is that all ICBM's have 10 RVs each. Also, since it is not considered important to assume particular distributions of ICBMs over available launch sites, a random distribution of the modeled missiles over these sites is used instead.

o Report generation - KEWSTAR has no capability for report generation. KEWSTAR/M now provides this function.

*See footnote page 13.

3.5.3 Inputs

The inputs for the Message Script Generator are given in Section 3.4.5.

3.5.4 Outputs

A single file, the Script file, is produced by the Script Generator. The Script file is designed to be the driving input to the network Simulator. The file organization is illustrated in Table 3.5.4.

Data Item	Denotes
Time	Reference, from start of simulation, when sighting occurs
Sensor ID	Unique node name
Sightings	Integer number of objects reported this period

Table 3.5.4 - Script file data description

4. COMMUNICATIONS NETWORK COMPONENT SIMULATOR

4.1 Introduction

The Communications Network Component Simulator is the heart of the algorithm testbed. Its purpose is to interpret data files which define the dynamic communications network status and connectivity. It models the communications network to the fidelity required for a realistic test of the communications network management algorithms that will be developed.

Section 4.2 (Functional Control and Data Flow) describes the overall operation of the component simulator. The simulator is viewed at a high level. Interfaces between the input files and the internal modules are described. Submodules of TLCSC 5 are described in detail. Section 4.3 (Node Level Description) describes the simulator submodels in greater detail. A model for communications node operation is described and explained. A high level functional description of the ALGUT is presented for the purpose of describing testbed interfaces.

4.2 Functional Control and Data Flow

The Component Simulator is used in Phase 2 of the simulation experiment, after the Simulation Experiment Planning phase has produced the four necessary run-time data files. Figure 4.2 illustrates how those four input files - Algorithm Parameters, Status, Configuration and Script are used to drive the Component Simulator.

Algorithms to be tested (ALGUTs) are "plugged in" to the node model represented in the diagram. The Algorithm Parameters file contains data used to set up and tune the ALGUT. That data varies according to the algorithm used.

Status and Configuration information affects all nodes. The Status Update function disables and enables nodes as the status scenario dictates. The Configuration Update function refreshes node positions as read from the Configuration file.

The Sensor Sightings Update function periodically informs the sensor nodes of the number of objects to be reported. This function is the only external stimulus to communications network loading. The Sensor Sightings and the Status Update functions comprise the available means of stressing the network.

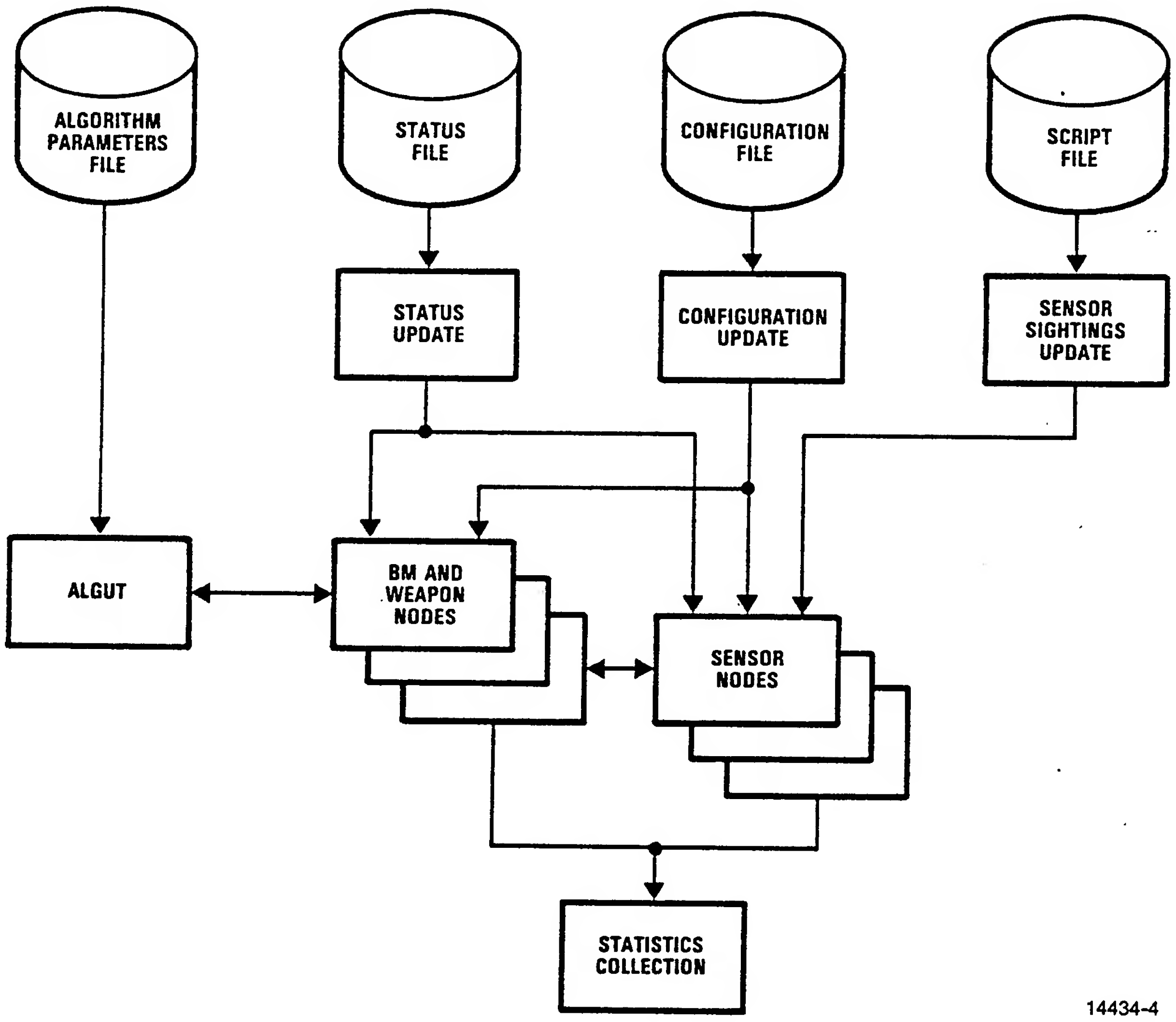


Figure 4.2 - Simulator Functional Control and Data Flow

4.2.1 Status Update

This function reads directives from the Status file and executes the instructions by enabling and disabling nodes and links as required. The algorithm is given in Figure 4.2.1.

```
read a status-time from the status file
if status-time is now then
     if type-status is temporary then
          disable node
          schedule a node-recovery
     end if
     if type-status is permanent then
          disable node
     end if
     if type-status is link-disable then
          disable link
          schedule a link-recovery
     end if
     read a status-time from the status file
else
     sleep until status-time
end if
```

Figure 4.2.1 - Status Update Algorithm

4.2.2 Configuration Update

This function reads position data from the Configuration file and updates node position coordinates. The coordinates are then used for line-of-sight and propagation delay computations. The algorithm is given in Figure 4.2.2.

```
read a config-time from the configuration file
if config-time is now then
       for each node
            update position
       end for
       update links
read a config-time from the configuration file
else
       sleep until config-time
end if
```

Figure 4.2.2. Configuration Update Algorithm

4.2.3 Sensor Update

This function notifies each sensor node of the number of sighted objects at each time step. The number is then used by the sensor node to generate sensor reports to the BM nodes. The Sensor Update algorithm is given in Figure 4.2.3.

```
read vis-time from the script file
if vis-time is now then
        for each sensor in file
                update number of objects sighted
        end for
        read vis-time from the script file
else
        sleep until vis-time
end if
```

Figure 4.2.3 - Sensor Update Algorithm

4.2.4 Statistics Collection

Statistics that are used to evaluate candidate ALGUT's are gathered by means of instrumentation inserted within the node and link submodels. A detailed description of the generalized SIMULA contexts to be used in the implementation is given in Reference 4.

4.3 Node Level Description and Data Flow

4.3.1 Node Overview

A complete node consists of three functional parts as illustrated in Figure 4.3.1. Those parts are OSI layers, network management routines (ALGUT's), and node recovery.

OSI layers are shells of standard protocol functions layered according to the ISO definitions. Five of the seven defined OSI layers are used here. The layers are not designed to provide generalized networking services, but include only those functions needed for this study. In fact, some of the functions shown may not be used for a particular simulation experiment. Ref. 8 provides an excellent discussion of the OSI model.

The network management function represents the set of implemented algorithms (ALGUTs) being evaluated in the test bed. Only an overview of these functions and their interface to the node submodel are discussed in this document.

Node recovery represents the design approach to temporary node outages due to EMP. Each node is assumed to have a capability to restore its data to a previous state.

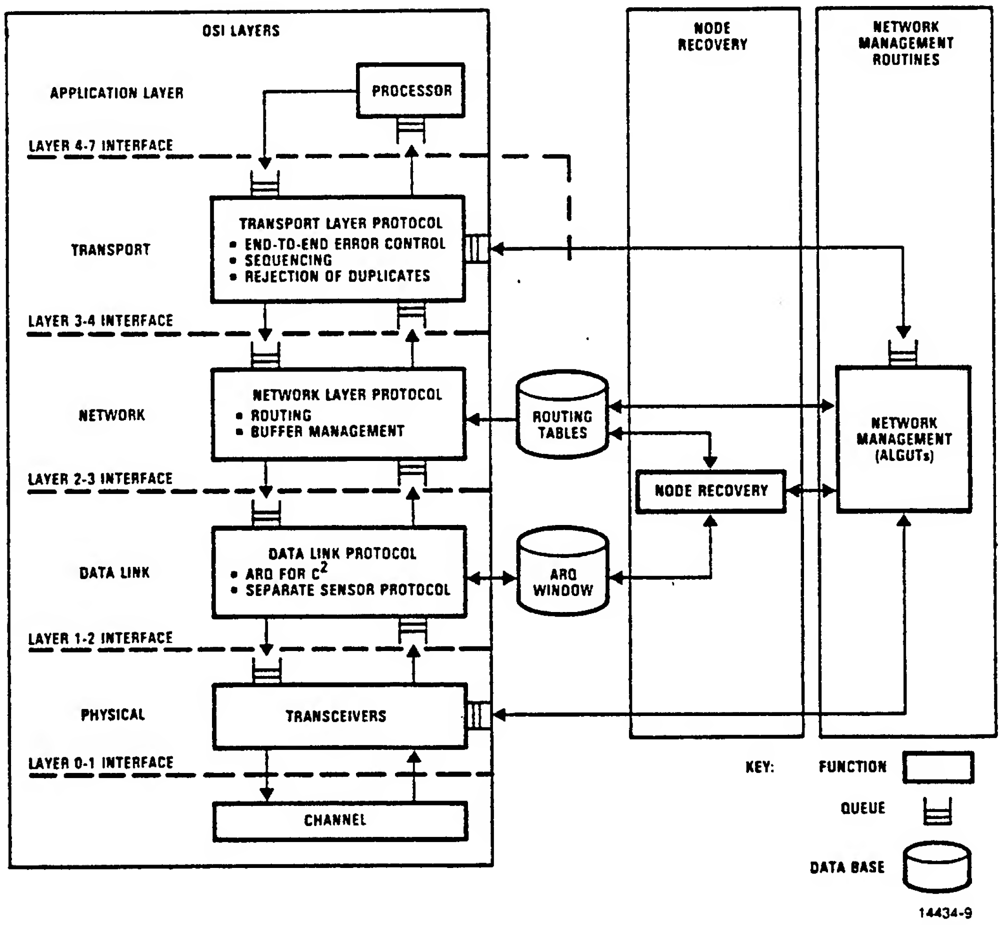


Figure 4.3.1 - Node Submodel Functional Control and Data Flow

4.3.2 OSI Layers

Five of the seven OSI model layers are represented to various degrees as shown in Figure 4.3.1. The session and presentation layers (Layers 5 and 6), are not considered to be important for purposes of this investigation. Each of the other five layers is described in greater detail below. Interfaces with other node elements are also described.

Interfaces between OSI layers are discussed in the informal NRL document "Description of Interfaces Between Protocol Layers". This paper, included as Appendix A, describes interfaces as they presently exist within the NRL testbed for OSI Layers 1-2 and 2-3. The description that follows builds upon that document.

o Physical (Layer 1): The physical layer includes transmission equipment used to place messages into the channel, the channel itself, and the receiving equipment. Messages at this level are perceived as strings of bits.

Two types of information pass between the physical layer and the ALGUT. Inputs to the physical layer are control signals that instruct a repointing function to aim the antennas to a different location in space. The control message is enqueued so that it may be acted upon in an orderly manner. The second type of information is associated with an ALGUT capability to sense the status of the physical layer. The exact nature of the status information needed depends upon the ALGUT.

o Data Link (Layer 2): Two data link layer protocols are needed for this investigation. An Automatic Repeat Request (ARQ) protocol is used to route C^2 messages, and a separate protocol is used for sensor messages.

The ALGUT has a capability for data link layer status sensing which is similar to the capability described for the physical layer.

Additionally, the ARQ window database, like all databases within the node, is periodically copied by the node recovery module. Node recovery does not maintain the database, but restores it in the case of an EMP occurrence.

o Network (Layer 3): Above the data link layer is the network layer. This layer includes a routing function whose purpose is to determine from routing tables which channel is appropriate for sending an outgoing message. The other function, buffer management, consists of arbitrating and scheduling the use of output channels.

The network layer contains interfaces with both the ALGUT and the node recovery function. Both interfaces take place through the routing tables. The routing tables database is considered to be a part of the network layer. The routing function can only access, but not update, the routing tables. Because of the highly dynamic nature of the logical and physical channels, and of the corresponding routing decisions, the maintenance of the routing tables database is one of the network management functions and a candidate ALGUT. Node recovery copies the database at intervals. Node recovery does not maintain the routing tables; it only restores them in the case of a memory wipe due to EMP.

o Transport (Layer 4): Several Component Simulator routines are contained within the transport layer. Three of them are error control, sequencing, and duplicate rejection. Error control is a method of controlling message error propagation through the network. Sequencing controls arriving message flow from the application layer according to priorities and arrival times. Rejection of duplicates shields the application layer from message overload due to the possible use of flooding or limited flooding schemes.

To the transport layer, the ALGUT is perceived as an application function. Arriving messages with a battle management content are passed to/from the application level processor; messages concerning network status and status requests are passed to/from the ALGUT.

o Application (Layer 7): The application level processor is viewed by the remainder of the network as a black box which is both a sink and a source for messages. The processor represents all the BM-related equipment at the node - the network user. Each type of node has a different type of processor which accepts and generates messages in a manner that is patterned after the intended function of the node. The flow control function is assumed to occur within the affected processors.

4.3.3 Node Recovery

Node recovery is a non-networking function that is provided as a part of the testbed to simulate node recovery from an EMP event. The box shown in Figure 4.3.1 has been expanded in Figure 4.3.3 to show data flow within the function. Two processes are active. The state saver periodically copies all other node databases and places the copies into the state database, which is presumed to be unaffected by EMP. The memory reload process operates only when the EMP actually occurs. It restores all databases to the last saved state so that node operation can resume.

4.3.4 Algorithm Under Test (ALGUT)

The ALGUT*, represented as a box in Figure 4.3.1, is not a part of the testbed. Instead, it is the reason for the existence of the testbed. It is described here for the purpose of explaining its interface with the Component Simulator. Figure 4.3.4 is an expanded view of the ALGUT internal and external data flow. The external arrows represent information passing across the interface between the ALGUT and its test bed.

Four major functions of the ALGUT are sketched in the following paragraphs. Details concerning these functions and their interfaces are beyond the scope of this document.

o Status Monitor - This function receives and sends information from/to other nodes concerning present network condition. Information needed by the Adaptive Routing Manager (ARM) and Resource Configuration Manager (RCM) defines the exact type of status message passed. Congestion and bottleneck locations are examples of monitoring data that may be required.

*ALGUT, or Algorithm Under Test, actually consists of a set of network management functions. Only one or a combination of these major functions might be under investigation in a particular simulation experiment.

o Topology Monitor - This function is concerned with physical connectivity changes, which may occur either because of unpredictable node or link outages, or because of predictable orbital dynamics. Messages are passed as necessary to the ARM and RCM.

o Adaptive Routing Manager (ARM) - This function is primarily responsible for maintaining the routing tables used to determine the logical path of an outgoing message through the network. Other tasks may include load balancing, providing of routing service categories for different messages, and adaptation of routes to physical connectivity changes.

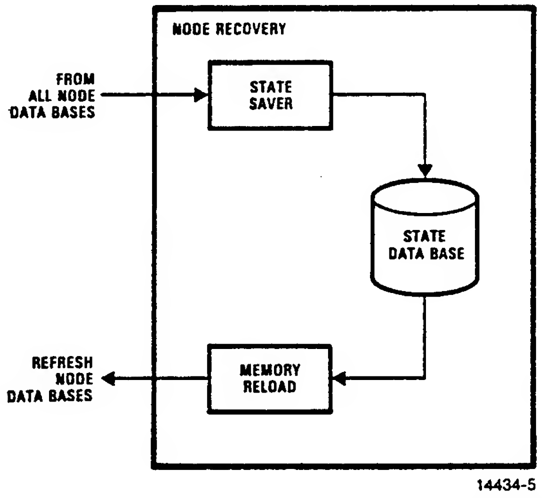


Figure 4.3.3 - Node Recovery Data Flow

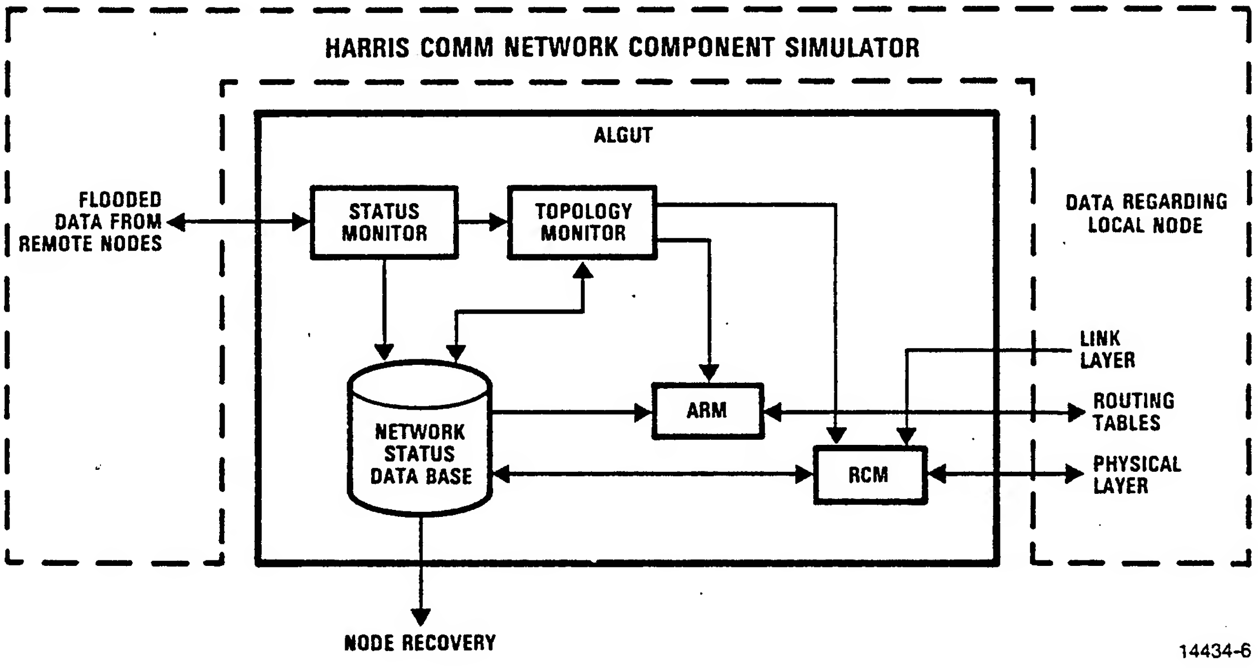


Figure 4.3.4 - ALGUT Data Flow

o Resource Configuration Manager (RCM) - This function is primarily responsible for the setup and maintenance of the physical network links. Links may be lost because of enemy-induced outages or because of orbital dynamics. Control signals are sent to receivers, transmitters, and antennas at the link layer for antenna repointing and link initialization. Successful establishment of new links is reported back to the RCM. Resulting connectivity changes are then reported in the network status database.

4.3.5 Interface Summary

Three types of interfaces are shown in Figure 4.3.1. Protocol interfaces are represented by the arrows that connect OSI levels. Node Recovery interfaces are represented by arrows connecting the Node Recovery function to other functions within the node. ALGUT external interfaces pass information between the network management algorithms and either the Communication Network Component Simulator or Generate Simulation Experiment (see Figure 2.3).

Table 4.3.5 summarizes information flow across the intra-nodal interfaces. Information flows from the function that the type interface is named after, to the named destination. A short description of the data follows in column three. If the same information flows backward across the interface, a check (✓) appears in the last column. If the return flow of information is dissimilar, no check is provided and the return information is briefly described.

Type of Interface	Information Flow		
	Destination	Description	Return
Protocol	All OSI Layers	Messages themselves Message arrival alerts	✓
Node Recovery	Routing Tables	All routing data	✓
	ARQ Window	All saved message data	✓
	Network Status Database	All network status information	✓
Network Management Algorithm	Transport Layer Protocol	Status messages Status requests	✓
	Routing Tables	Updates	Routing data
	Physical Layer	Antenna repointing commands	Repointing completed signal Sense status
	Data Link Layer	Link quality data	Sense status

Table 4.3.5 - Intra - Nodal Interfaces Summary

5. REFERENCES

Applicable references include the Request for Proposal (RFP) for this contract and several papers which describe software incorporated into the Harris Simulatior design. Two readily available books which contain supporting background information for procedures and methods used complete the reference list.

5.1 RFP

1. Naval Research Laboratory solicitation number N00014-85-R-MM51, 14 August 1985.

5.2 Papers Describing Software Used

2. "Layer 1: A SIMULA Context for Simulating the Operation of Communication Systems", Hauser, James P. and Baker, Dennis J., unpublished.

3. "An Event-Process Facility Built on SIMULA: A Tool for Simplifying the Simulation of Distributed Control Systems", Baker, Dennis J. and Hauser, James P., unpublished.

4. "An Abstract Type for Statistics Collection in SIMULA", Landwehr, Carl E., ACM Trans. on Programming Languages and Systems, Vol 2, No. 4, October 1980, pp. 544-563.

5. "A System Parameter Model of Space-based Kinetic Energy Weapons in Ballistic Missile Defense, Master's thesis", Capts. Chapman, Daniel W. and Ring, Richard L., document #AFIT/GSO/GOR/OS/84D-1, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio.

6. "Monthly Technical Status Report #1 for Adaptive Distributed Network Management System", Harris Government Systems Special Programs Office, Melbourne, FL, 20 March 1986.

5.3 Books

7. Fundamentals of Astrodynamics, Bate, Roger B., Mueller, Donald D., and White, Jerry E., Dover Publications, Inc, New York, 1971.

8. Computer Networks, Tanenbaum, Andrew S., Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1981.

APPENDIX A

Author: Ed Althouse, NRL

Description of Interfaces Between Protocol Layers

Between each set of protocol layers there are four basic procedure calls which provide the interface function. There will also be additional auxiliary procedure calls that will vary from interface to interface. We will first specify the four basic procedures and then discuss the auxiliary procedures that depend on the specific layer boundary.

Message-passing down to lower level

This is accomplished by a simple "send" call in the form of

send(n_msg)

where n_msg is a pointer to the message in the upper layer (which we denote as layer "n"). To provide ample generality for all situations, we buffer the message transfer across the layer boundary in both directions. Consequently, when the message is sent down to the lower layer (which we denote as "n-1"), it is immediately buffered (or put into a queue) so that the "send" command can be requested as frequently as desired and without any restrictions resulting from the state of the lower layer protocol. The "put(msg)" procedure found in the CONTEXT class msg_queue is used for the queuing function. Normally, the type of message changes at each interface; this corresponds to the adding or stripping off of header/trailer information as the message works its way down or up the layered protocol. For example, layer 7 will send down an application message, layer 3 will send down a net message, layer 2 will send down a link message, and layer 1 will send out a channel message.

If the message cannot be delivered to the lower layer (the actual reasons for no delivery may be protocol dependent), the second interface procedure

msg_returned(n_msg)

is called to return the message to the upper layer and to simultaneously provide a notice that the send request failed.

Message-passing up to a higher level

As previously stated, message transfer across the layer boundary will be buffered in both directions. Consequently, we assume that in the upward direction, the message will be put into a queue in the "n-1" layer. The introduction of the message into the queue is used as a stimulus to send a

msg_received(n_msg)

notification across the interface which results in an interrupt condition in layer "n". Note that n_msg is a pointer to the "n" layer message rather than the "n-1" layer message. The protocol in layer "n" uses the interrupt notification to request the procedure get(n_msg,cond)

which extracts the message from the queue and passes it up to the higher layer subject to the value of the boolean parameter "cond". (The get procedure is found in CLASS msg_queue. Normally "cond" is set to TRUE in the procedure call which results in removing the oldest member of the queue (top of the queue) and making it available. Condition can, alternatively, be set to some expression, such as msg.id_num = 5, which will result in searching through the queue and removing the message with attribute id_num=5.

The remainder of the discussion will be devoted to discussion of the interface procedures that are specific to particular boundaries.

1-2 Interface

At the 1-2 interface, there are a large number of tasks that must be handled such as selecting transmitters, receivers, and channels; setting frequencies or frequency-hop codes; detecting collisions; setting transmission rates; turning transmitters on and off; etc. These procedures are documented in "Layer 1: A SIMULA Context for Simulating the Operation of Communication Systems -- J.P. Hauser and D.J. Baker". Some of the important procedures are:

```
select_xmtr(xid) -- (select transmitter numbered xid)
select_rcvr(rid) -- (select receiver numbered rid)
target_rcvr(node_target) -- (point the selected receiver at the targeted
                             node)
target_xmtr(node_target) -- (point the selected transmitter at the
                             targeted node)
start(msg) -- (procedure call to start the activity of the controller,
   transmitter, comm. link, etc. associated with sending the message
   having the pointer "msg")
```

2-3 Interface

At the 2-3 interface, there is the necessity to relate layer 3 routing destinations with layer 2 link_id's. These functions are served (on the way down) by the procedure

select_node(nid)

which selects the node that is to receive the message) and (on the way up) by the procedure

node_rcvd_from

which identifies the node that sent the message. Down in layer 2 there will be an association between a transmitter number and the destination node targeted by select_node(nid) as well as an association between a receiver number and the node that sent the message.